Detecting intrusion with rule-based integration of multiple models

Sang-Jun Han and Sung-Bae Cho

Department of Computer Science, Yonsei University, 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea {sjhan, sbcho}@ cs.yonsei.ac.kr

Abstract

As the information technology grows interests in the intrusion detection system (IDS), which detects unauthorized usage, misuse by a local user and modification of important data, has been raised. In the field of anomaly-based IDS several data mining techniques such as hidden Markov model (HMM), artificial neural network, statistical techniques and expert systems are used to model network packets, system call audit data, etc. However, there are undetectable intrusion types for each measure and modeling method because each intrusion type makes anomalies at individual measure. To overcome this drawback of single-measure anomaly detector, this paper proposes a multiple-measure intrusion detection method. We measure normal behavior by systems calls, resource usage and file access events and build up profiles for normal behavior with hidden Markov model, statistical method and rule-base method, which are integrated with a rule-based approach. Experimental results with real data clearly demonstrate the effectiveness of the proposed method that has significantly low false-positive error rate against various types of intrusion.

Keywords: intrusion detection systems, anomaly detection, multi-measure modeling

Introduction

The recent advances of computer communication infrastructure realizes the era of computer-based data processing. However, the higher reliance on computers becomes, the more concerns about computer security have been raised. According to a report from CERTCC-KR in 2000, the number of security incidents has increased 3 times more than that in 1999 and 5,333 incidents have reported in 2001 [1]. In addition, the high availability of hacking tools due to the proliferation of the internet causes attacks simply out of curious.

As the damage from security incidents increases, several tools become essential to computer systems. Intrusion detection is to find attacks exploiting illegal uses or misuses. An IDS is software to detect attacks exploiting illegal uses or misuses and modification of important data by analyzing system calls, system logs, activation time, and network packets of each operating system [2]. Generally, intrusion detection techniques can be divided into two groups according to the type of data they use: misuse detection and anomaly detection [3].

Misuse detection uses knowledge about known attacks and attempts to match current behavior against the attack patterns. It has the advantage that known attacks can be detected reliably with low false-positive error and economically. The shortcoming is that it cannot detect unknown attacks. To remedy the problem of detecting unknown attacks, anomaly detection attempts to detect intrusions by noting significant deviation from normal behaviors. It has low false-negative error rate where attacks are considered as normal behaviors because the behaviors deviating from normal behavior are considered as intrusion. However, it suffers from high false-positive error rate because unseen normal behaviors are considered as attacks. Another drawback of anomaly detection technique is that the detectable types of intrusion are limited according to the measures and modeling methods used.

In this paper, to overcome drawbacks of the conventional anomaly detection techniques, we propose an anomaly-based detection technique that uses multiple measures and models. First of all, we develop four appropriate detection methods that use system call events, resource

ELSEVIER

Computers & Security
Vol 22, No 7, pp 613-623, 2003

0167-4048/03

usage of process, file access events as the measure of normal behavior, with the three modeling methods. Second, a rule-based approach is proposed as a technique to integrate them. The proposed detection method is expected better performance because it can model normal behaviors from various perspectives.

The rest of this paper is organized as follows. In Section 2, we give a brief overview of the related works. The overall design and detailed description of the proposed methods are presented in Section 3. Experimental results are shown in Section 4.

Related works

Various data mining techniques have been applied to intrusion detection because it has the advantage of discovering useful knowledge that describes user's or program's behavior from large audit data sets. Statistics, artificial neural network and HMM (hidden Markov model), rule learning, outlier detection scheme are some of the data mining techniques widely used for anomaly and misuse detections. The representative studies on intrusion detection are summarized in Table 1.

Statistics is the most widely used technique, which defines normal behavior by collecting data relating to the behavior of legitimate users over a period of time [4]. To detect an intrusion, it determines whether current behavior is deviated from the normal behavior more than the threshold based on the standard deviation. NIDES (Next-generation Intrusion Detection Expert Systems) is the representative IDS based on statistics that measures the similarity between a subject's long-term behavior and short term behavior for intrusion detection [7]. The detection rate is high because it can use various types of audit data and detect intrusion based on the previous experimental data. The shortcoming is that it is not sensitive to some behavior and detectable types of intrusion are limited.

Hyperview of CS Telecom is a representative IDS using neural network [9]. It consists of 2 modules: neural network and expert system. The neural network in Hyperview uses temporal sequence of audit data as inputs and 60 types of audit data: CPU usage, memory usage, etc. R. Lippmann et al. have applied neural network to keyword-based detection system [12]. They have used keyword counts from transcripts for telnet session as inputs of

Table 1. The representative studies on intrusion detection
ES: Expert System, NN: Neural Network, ST: Statistics,
HMM: Hidden Markov Model, RL: Rule Leaning, OD: Outlier Detection

Organization	Name	Period	Technique					
			ES	NN	ST	HMM	RL	OD
UCDavis	NSM [5]	1989-1995	X		X			
SRI International	IDES [6]	1983-1992			X			
	NIDES [7]	1992-1995	X		X			
	EMERALD [8]	1996-	X		X			
CS Telecom	Hyperview [9]	1990-1995		X	X			
Univ. New Mexico	C. Wrannder et. al [10]	1995			X	X	X	
Yonsei Univ.	Park and Cho [11]	1999-				X		
MIT Lincoln Lab.	R. Lippmann et. al [12]	1999-		X				
Colombia Univ.	MADAM ID[13]	1998-					X	
Univ. of Minnesota	A. Lazarevic et. al [14]	2002-						X

neural network. While the artificial neural network has some similarity to statistical techniques, it has the advantage of easier representation of nonlinear relationship between input and output. The defects of neural networks are that its computational load is very heavy and it is difficult to interpret the relationship between inputs and outputs.

An HMM is a useful tool to model the sequence of observed symbols of which the construction mechanism cannot be known. The representative one is the technique proposed by C. Warender of New Mexico University [10]. It uses system call audit trails to measure normal behaviors. While HMM produces better performance in modeling system call events than other methods, it requires a very long time for modeling normal behaviors. The solution for this problem might be to improve the performance of the computer system or to reduce the training data. The technique that reduces audit trails by filtering audit trails from abstracted information around the change of privilege flows can save the computational costs significantly while maintaining good performance [11].

RIPPER [15], a rule learning tool, has been used for automatic construction of detection models. RIPPER is applied to labeled training data sets and automatically mines the patterns of intrusion in MADAM ID [13]. Though it is good tool for discovering patterns, anomaly detection technique is required for the detection of novel intrusions.

The outlier detection scheme which is one of the data mining techniques attempts to identify a data point that is very different from the rest of the data. A. Lazarevic et al. have applied it to anomaly detection. They have compared several variations of outlier detection algorithm and in their experiment, local outlier factor (LOF) approach shows the best performance against 1998 DARPA intrusion detection evaluation data [14].

Table 2. Recent trends of attacks

	May 2002	June 2002	July 2002
Buffer overflow	34	25	10
Control, setup vulnerability	3	4	0
Denial of service	0	5	0
S/W security vulnerability	0	1	1
Total	36	30	10

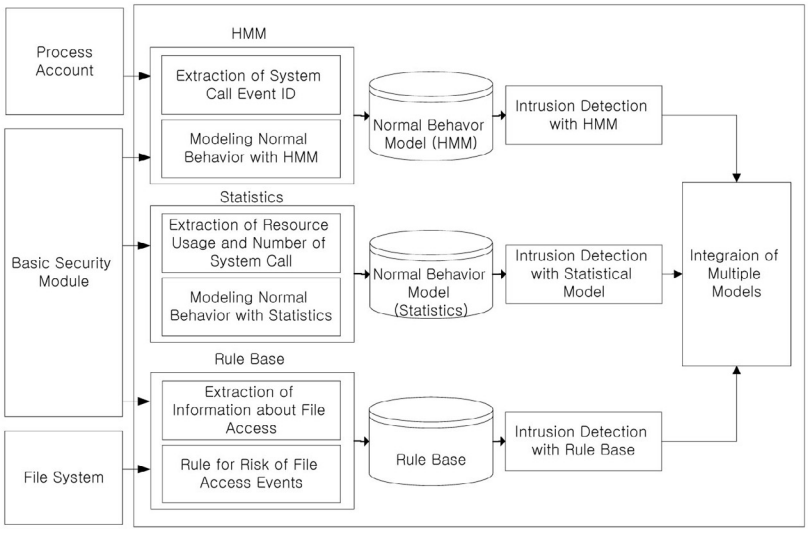
As we have seen, the techniques that are currently used for intrusion detection have their own pros and cons. This paper attempts to find better intrusion detection method with the integration of several data mining techniques which use different kind of methods to improve performance of intrusion detection system.

Multiple measures and models integration

Among the intrusion types that frequently occur, buffer overflow, S/W security error, configuration error and denial of service attacks are prevalent on host. The hacking trends in May, June, and July 2002 provided by CERTCC are given in Table 2. As shown in the table the majority of attacks are buffer overflow. Recently, massive access to the internet raises the issue of denial of service attack. This paper focuses on the two intrusion types to develop sophisticated detection method.

In this paper, we use common measures for host-based detection system: system call event, file system information and resource usage of process. Though there are several methods for modeling each measure, we select modeling methods appropriate to each measure considering the relationship between the characteristics of intrusion traces and the power of modeling methods. However, because each intrusion type leaves anomalies at individual measure, there are undetectable intrusion types in each measure and modeling method. To overcome this drawback, it is necessary to remedy shortcomings of each method through integrating the results of several methods. We

Figure 1. The architecture of the proposed method



construct the rules to combine multiple measure models and detect intrusions with them. The architecture of the proposed method is shown in Figure 1.

HMM with system call events

Sun Microsystem's Basic Security Module (BSM) which is an auditing facility for Solaris provides an adequate representation of the behavior of the program because any privileged activities that might be generated by a program are captured by BSM. Usually an audit trail from BSM consists of several measures. A system call, one of the measures from BSM, can be either perfectly normal or dangerous, depending on the situation. For example, the program attacked by buffer overflow generates system call events that are significantly different from events generated in a normal situation. Thus, we can detect intrusions effectively by building the model of system call events from normal situation and noting significant deviation from the model. In this paper, for modeling system call event sequences, we use HMM that is widely used for speech recognition because it is very useful for modeling sequence information. HMM can be successfully applied to modeling system call event sequences [16].

A model λ is described as $\lambda=(A, B, \pi)$ using its characteristic parameters as follows: an HMM is characterized by a set of states Q and a set of possible symbol observations V, the number of observation symbols M, state transition probability distribution A, observation symbol probability distribution B and initial state distribution π.

The set of all system call events in audit data corresponds to the set of possible symbol observations V and the number of events corresponds to M. Audit trail are sampled into fixed-length windows with each window sliding one event at a time because the whole audit data cannot be applied to HMM at once. The length of observation sequence T corresponds to the length of window. The type of HMM that we use is a left-right model, which is known for modeling temporal signal better than other models [17].

Intrusion detection with HMM consists of two phases: normal behavior modeling and anomaly detection. The first phase is normal behavior modeling, which is determining HMM parameters to maximize the probability $\Pr(O|\lambda)$. Because no analytic solution is known for it, an iterative method called Baum-Welch reestimation is used. Anomaly detection, the second phase, matches current behavior against the normal behavior model, and calculates the probability with which it is generated out of the model. Forward-backward procedure is used for this purpose [18]. The probability is used to decide whether normal or not with a threshold.

Statistics with resource usage and system call events

Most Unix-based operating systems serve Process Account (PACCT) as audit data. It provides the resource usage of processes: CPU time, memory usage, I/O usage, etc. Denial of service (DoS) attack sharply raises the resource

usage of victim process. This type of attack can be detected by noting processes that show unusual resource usage compared with normal behavior using PACCT audit data.

PACCT audit data is useful to detect DoS attacks. Unfortunately, we cannot detect attack type that targets resource not recorded to PACCT. For example, attack that consumes process table leaves no anomalies in PACCT. However, it unusually generates large amount of system call events. In this case, we can detect that by noting process that generates unusual number of system call events.

In this paper, we use statistical techniques to model normal resource usage and the number of system call events. This statistical approach is a modified method that has been used in NIDES. For each audit record generated by a user, we generate a single test statistic value denoted by T that summarizes the degree of abnormality in the user's behavior in the near past. Large values of T indicate abnormal behavior, and values close to zero indicate normal behavior. In the case of PACCT, the T statistic is a summary judgment of the abnormality of measures in PACCT. We denote the individual abnormality of measures by S, each of which measures the degree of abnormality of behavior with respect to specific features such as CPU time, memory usage and I/O usage. T statistic has been set equal to the weighted sum of the S statistics as follows:

$$T = a_1 s_1 + a_2 s_2 + \ldots + a_n s_n$$

where s_n is the S score of each measure and a_n is the weight to each measure.

Each S statistics is derived from a corresponding statistic called Q. In fact, each S statistic is a normalizing transformation of the Q statistic so that the degree of abnormality for different types of features can be added on a comparable basis. The value of Q corresponding to the current audit record represents the number of audit records that are arrived in the recent past. In order to transform Q to S, we have built a normal historical profile of all previous values of Q and compare the current value of Q with this normal profile to determine if the current value is anomalous.

The normal profile of Q is categorized into intervals of values and we build historical relative frequency with which Q belongs to each interval. Next, we obtain the tail probability of this frequency, and transform the tail probability to S statistic with cumulative normal distribution function. S statistic would be a large positive value whenever the current Q statistic was in the interval that has low frequency because this is a relatively unusual value for Q or whenever Q was not in any interval because this has not historically occurred. For example, if the current Q statistic is in the interval whose frequency is the 80th percentile (i.e., the tail probability is 20%), then the corresponding value for S is 1.28. The selection of appropriate intervals is important for functioning of the algorithm. In this paper, 9 intervals for categorizing Q are used for each Q measure. We have defined the interval spacing with regard to a distribution of Q generated by normal behavior.

Small value of Q indicates a recent past that is similar to historical behavior, and large values of Q indicates a recent past that is not similar to historical behavior. Given k is an index of appropriate audit records, t_k is the time that elapses between the kth and most recent audit records, r is the decay rate and D_k is the change that occurs between the (k+1)st and kth appropriate audit records, Q is defined as follows [19]:

$$Q = \sum_{k \geq 1} D_k \times 2^{-rt_k}$$

The decay rate r determines the half-life of the measure Q. The half-life specifies the number of audit records or days of audit record activities that constitute short-term and long-term behaviors. Large values of r imply that the value of Q will be primarily influenced by the most recent few appropriate audit records. Small

values of r imply that Q will be more heavily influenced by audit records in the more distant past. For example, a half-life of 10 minutes corresponds to r value of $0.10 = log_2(0.5)/10.0$ if time is measured chronologically.

The elapsed time t_k between audit records can be measured chronologically or in terms of appropriate audit record counts. If time is measured chronologically, the strength is that Q tends to concentrate on activity in the past few hours. The weakness is in instability because Q depends on a variable number of audit records. If time is measured in terms of audit record counts, a principal strength of Q is stability because Q becomes based on what is essentially a fixed number of audit records. The weakness is potentially extended or compressed time frame of past activity that affects the value for Q. In this paper, we measure time using audit record counts because the usage of computer system used to generate normal behavior is irregular.

Figure 2. No Read Up rule

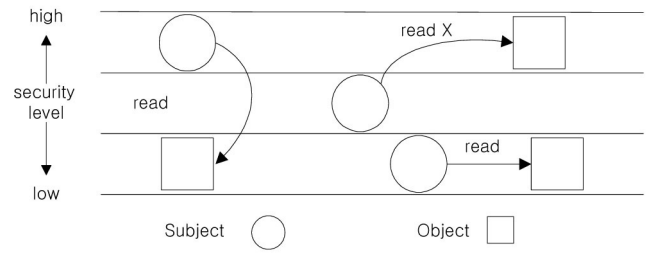


Table 3. Examples of file access risks

Subject	Object	Mode	Risk
root	root	755	5
root	user	755	5
user	root	744	9
user	root	755	14
user	admin	644	4
user	guest	555	3
guest	root	711	17
guest	admin	555	6
guest	user	755	6

Rule base with file access events

Generally, attacks tend to access files of which the security level is high. For example, it executes file which has a SETUID privilege to acquire root privilege or attempts to read files owned by root to destroy the computer system or obtain a critical data. Owing to this tendency of abnormal behavior, it is adequately suspicious behavior for ordinary user to access files which have higher security level.

The best-known security policy related with file access is the Bell-LaPadula (BLP) model which has been formulated by Bell and LaPadula [20]. In an access to some information, there are three primary elements: subject, object and access attribute. The subject corresponds to user or program, the object corresponds to files and the access attribute corresponds to the kind of access model: read, write, execute, etc. In order to determine if a specific access mode is allowed, the BLP model compares the clearance of a subject with the classification of the object and determination is made as to whether the subject is authorized for the specific access mode [21]. The BLP model includes several rules referenced frequently. One of them is no read up rule, which allows access if the current level of the subject dominates the level of the object as shown in Figure 2. It prevents user from accessing information for which they are not allowed to access.

In this paper, we audit file access and evaluate the risk of access event by analyzing the content of that. The security levels of subject and object are divided into 4 categories: root, administrator, ordinary user and guests. The risk of access event is evaluated to one of 21 levels according to the difference of security levels between subject and object. Some resultants of evaluating risks are showed in Table 3.

The access event of which the difference between two levels is bigger has higher risk. If the level of object is lower than that of subject the risk is not increased due to the security level

difference. If the access event contains a file that allows more operations, it has a higher risk. For example, when the file mode is 755 the access event is riskier than that of 744.

Integrating multiple measure models

In this paper, we have proposed an intrusion detection technique that integrates detection methods in order to increase detectable attack types and reduce false-positive error rate. We use rule-based approach for integration. The rules are made experimentally considering the relationship between the characteristics of intrusion traces and the capability of modeling methods. The rules that used for integration are as follows:

- IF ((HMM/System Call > T1) AND (Rule-base/File Access < T4))
  THEN Buffer overflow attack
- IF ((Statistics/Resource Usage > T2) AND (Statistics/System Call < T3))
  THEN Consumption of Process Table (DoS attack)
- IF ((Statistics/System Call > T3) AND (Statistics/Resource Usage <T2))
  THEN Consumption of Disk or Memory (DoS attack)

T1: Threshold of HMM/System Call,
T2: Threshold of Statistics/Resource Usage,
T3: Threshold of Statistics/System Call and
T4: Threshold of Rule-base/File Access.

The first rule integrates HMM with system call event and rule-based method with file access event. This rule prevents HMM from considering unseen normal behaviors as attacks. We can reduce false-positive error rate with this rule because most of buffer overflow attacks attempt to access file that is owned by root or has SETUID privilege.

The second rule integrates statistics with resource usage and statistics with system call event. Statistics with system call event has difficulty in considering normal behavior that uses many system calls. An attack that results in denial of service by excessive amount of system call events shows low resource usage while the number of events it generates is very large. This rule can reduce error rate by considering as attack the behavior that generates unusual number of system call events although resource usage is normal.

The last rule also integrates statistics with resource usage and statistics with system call event. Statistics with resource usage has difficulty in considering normal behavior that requires much resource. An attack that results in denial of service by consumption of limited resource shows a small number of system call events while the resource usage is very high. This rule can reduce error rate by considering behavior as attack, which shows resource usage although it generates usual number of system call events.

Experiments

We have collected normal behaviors from six graduate students for two weeks using the Solaris 7 operating system. They have mainly used text editor, compiler and programs of their own writing. Total 13 megabytes (160,448 records) of BSM audit data and 840 kilobytes of PACCT audit data have been collected from 16,470 commands. We also collect audit data that contain labeled attacks for testing in the same operating system. It contains 9 cases of u2r buffer overflow intrusion and 4 cases of denial of services. The attacks used in our experiments are as shown in Table 4.

Table 4. The attacks used in our experiments

Attack Type	Attack Name
Buffer Overflow	kcms_configure buffer overflow vulnerability lpset -r buffer overflow vulnerability xlock heap buffer overflow vulnerability
Denial of Service	Consumption of Disk Consumption of Process Table Consumption of Memory

Performance metrics

The important metrics in evaluating the performance of IDS are detection rates and false-positive error rate. In this paper, we apply common measure to each detection method as follows:

$$\text{detection rate} = \frac{\text{the number of processes that are real intrusion and considered as intrusion}}{\text{the number of processes that are real intrusion}}$$

$$\text{false} - \text{positive error rate} = \frac{\text{the number of processes that are normal but considered as intrusion}}{\text{the number of processes that are normal}}$$

In the experiments, we visualize the performance of detection method through ROC (Receiver Operating Characteristics) curve. An ROC curve depicts the changes in attack detection rate and false-positive error rate as modifiable parameter is changed. In this paper, evaluation threshold is used as a modifiable parameter. A desirable intrusion detection system must show a high detection rate at low false-positive error rate. In ROC curve, top-left curve is more desirable.

Figure 3. ROC curve with respect to HMM parameters

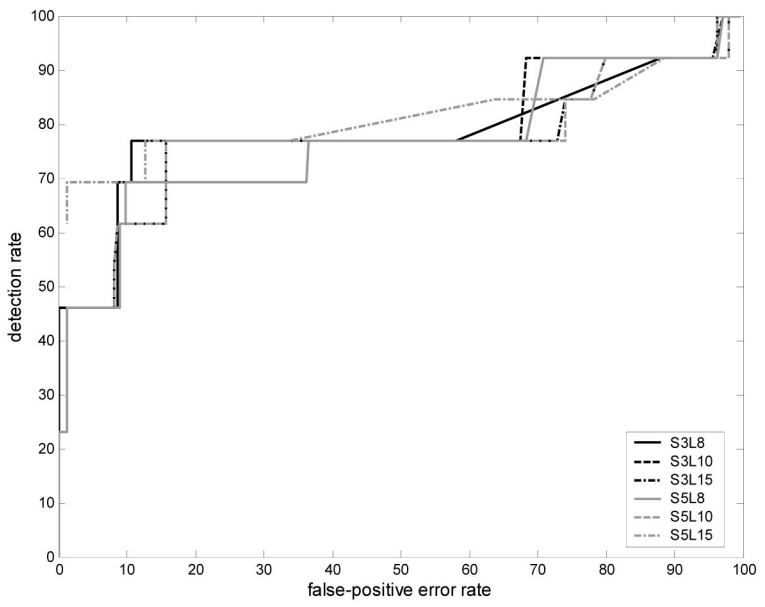


Table 5. The anomaly detection rate of each HMM parameter

State/ sequence	Threshold	Detection rate	F-P error rate	d'
3/8	-17.6	77%	10.699%	1.979
3/10	-9.7	77%	36.626%	1.078
3/15	-16.7	77%	15.638%	1.746
5/8	-20.5	77%	12.757%	1.874
5/10	-9.7	77%	36.626%	1.078
5/15	-16.6	77%	15.637%	1.746

We use discriminability to measure numerically the performance of a detection methods and efficiency to compare integration methods with others. The discriminability is a measure of the average intensity difference perceived by an observer between samples including a signal and samples not including a signal. It has higher value at high detection rate and low false-positive error rate [22]. Generally, it has been noted as d' and defined as follows:

$$d' = z(H) - z(F)$$

where z is the inverse of the normal distribution function, H is detection rate and F is false-positive error rate

The efficiency is used to evaluate the performance of the proposed integrated method. If the value of efficiency is over 1, we can say that the proposed method improves the performance. It has been noted as E and defined as follows where d_A' is the discriminability of detection method, and d_B' is the discriminability of the proposed method:

$$E = \left(\frac{d_A'}{d_B'}\right)^2$$

Experimental results

(1) Intrusion detection with individual methods

At first, we have conducted experiments without the integration of detection methods in order to identify the characteristics of each method. We have performed experiments to decide the optimal HMM parameters for effective intrusion detection. Figure 3 shows an

ROC curve according to the number of states and the length of sequences.

Optimal HMM parameters are obtained: the number of states is 3 and the length of input sequence is 8. However, at a detection rate higher than 77%, the false-positive error rate is sharply raised because HMM with system call event cannot expose intrusions well except for buffer overflow. Since denial of service attacks consisted of abuse of a perfectly legal action, it did not show any abnormality when we analyzed system call event. We compare the results from each parameter setting when the detection rate is 77% because it shows a reasonable false-positive error rate, as shown in Table 5.

We have made experiment of statistics with resource usage with changing the weight ratio. The sum of weights is fixed to 3 and we distribute the weight to each measure with an appropriate ratio. For example, when the weight ratio is 2:2:1, the real weight values are 1.2, 1.2 and 0.6. We have compared the results when changing the weight ratio at the same detection rate, as shown in Table 6.

The optimal weight ratio is 2:1:2 to CPU time, memory usage and I/O usage because this ratio shows the highest value of d'. Unfortunately, the false-positive error rate has sharply raised when detection rate is more than 23%. It indicates that the capability of this method is limited in some denial of services.

The experimental result of statistics with system call event is given in Table 7. The denial of service attack by the consumption of process table is exposed well because it generates excessive amount of 'fork' system calls. However, the performance is not reasonable because the number of system call events is not significantly different from the number of normal ones.

Table 8 shows the result of rule base with file access event. Because most of the attacks which aim to get root privilege attempt to access the file that has high security level. However, the false-positive error rate is sharply raised when the detection rate is more than 69.2% because the denial-of-service attacks disable the computer system without giving access to files of a high security level.

Table 6. The anomaly detection rate of each weight ratio

Weight	Threshold	Detection rate	F-P error rate	d'
2:1:1	7.0	23.0%	4.527%	0.956
1:2:1	5.2	23.0%	5.350%	0.876
1:1:2	7.1	23.0%	2.881%	1.162
1:2:2	6.0	23.0%	2.881%	1.162
2:1:2	7.4	23.0%	2.469%	1.229
2:2:1	6.0	23.0%	4.527%	0.956

Table 7. The anomaly detection rate of statistics with system call event

Threshold	Detection rate	F-P error rate	d'
10	30.77%	4.1152%	1.235
20	53.85%	7.4074%	1.543
30	76.9%	13.1687%	1.855

The experimental result of single detection methods does not show good performance owing to the characteristics of measure and modeling methods. We have compared the detection methods with the best parameters using ROC curves. The false-positive error rate has been sharply raised after a certain degree of detection rate as shown in Figure 4.

(2) Intrusion detection with integrated method

This experiment is conducted with a method that integrates detection methods. We have

Table 8. The anomaly detection rate of rule base with file access event

Threshold	Detection rate	F-P error rate	d'
13	23.0%	0.000%	2.354
11	46.2%	0.000%	2.994
9	69.2%	15.226%	1.529
7	69.2%	31.687%	0.979
5	69.2%	46.091%	0.601

Figure 4. ROC for individual detection methods with the best parameters

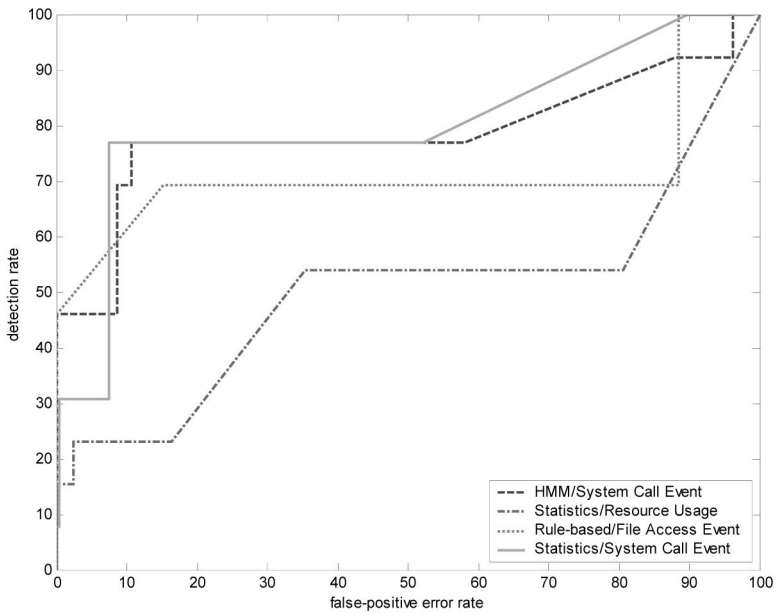
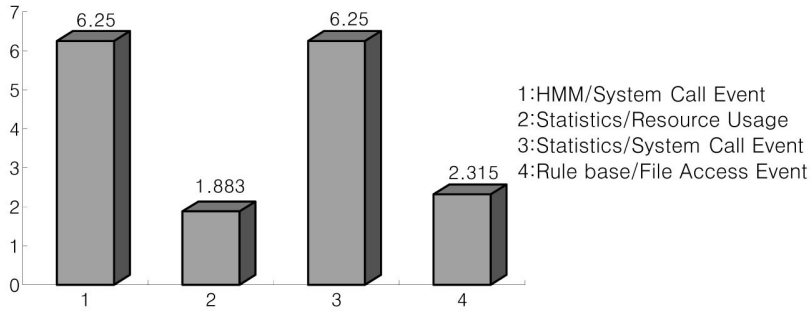


Table 9. A comparison of each detection method

	Detection rate	F-P error rate	d'
HMM/System Call Event	100%	99.177%	1.866
Statistics/Resource Usage	100%	80.658%	3.400
Statistics/System Call Event	100%	99.177%	1.866
Rule base/File Access Event	100%	88.477%	3.066
Integration	100%	5.761%	4.665

Figure 5. The efficiency of integrated method



used the parameter of each method that shows the best performance in the previous experiments: The number of states is 3 and the length of input sequence is 8 in HMM, the weight ratio to resource usage is 2:1:2 (CPU time: memory usage: I/O usage). The thresholds used are -17.6 to HMM, 6.0 to statistics with resource usage, 30 to statistics with system call event and 10 to rule base with file access event. We have compared false-positive error rate of each method at 100% detection rate as shown in Table 9.

The result shows the integrated detection method dramatically reduces the false-positive error rate. We have compared the improvement performance with *E* value. As shown in Figure 5, the integrated method shows better performance than any of the other detection methods because the rules alleviate the shortcomings of each method.

Conclusions

In this paper, we have presented effective modeling methods for three audit data and proposed a novel intrusion detection technique that integrates the detection methods. The proposed method uses multiple measure and modeling methods and integrates the results of individual detection methods with rule-based approach to overcome the drawbacks of the conventional anomaly detection techniques.

We evaluate the performance of each detection method and compare the results with integrated method. The integrated method shows 5.761% false-positive error rate at 100% detection rate whereas each element method produces more than 80% false-positive error rate at the same detection rate. It indicates that we can overcome the drawbacks by integrating several methods. This approach can be strengthened by adopting better element methods and integration technique with state-of-the-art data mining techniques.

However, we cannot guarantee that the rules used are optimal because they are made empirically and attack types that are not

expressed in the rules cannot be detectable. Machine learning techniques, such as decision tree and neural network, can be used for the integration to remedy this shortcoming. In the future, it is also needed to develop other measure and modeling methods to increase detectable attack types.

Acknowledgements

This research was supported by University IT Research Center Project.

Reference

[1] CERTCC-KR, Security Incident Statistic in Korea, http://www.certcc.or.kr/english, 2002.

[2] H.S. Vaccaro and G.E. Liepins, "Detection of anomalous computer session activity," *In Proceedings of IEEE Symposium on Research in Security and Privacy*, pp. 280-289, 1989.

[3] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," *Technical Report 99-15*, Department of Computer Engineering, Chalmers University, March 2000.

[4] E. Biermann, E. Cloete and L. M. Venter, "A comparison of intrusion detection systems," *Computers & Security*, vol 20, no. 8, pp. 676-683, December 2001.

[5] T. Heberlein, G. Dias, K. Levitt, B. Mukherjee, J. Wood and D. Wolber, "A network security monitor," *In Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*, pp. 296-304, Los Alamitos, CA, USA, 1990.

[6] T.F. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, C. Jalali, and P.G. Neuman, "A real-time intrusion-detection expert system (IDES)," *Technical Report Project 6784*, CSL, SRI International, Computer Science Laboratory, SRI International, February 1992.

[7] D. Anderson, T.F. Lunt, H. Javits, A. Tamaru and A. Valdes, "Detecting unusual program behavior using the statistical components of NIDES," *NIDES Technical Report*, SRI International, May 1995.

[8] P.A. Porras and P.G. Neumann, "EMERALD: Event monitoring enabling responses to anomalous live disturbances," *In Proceedings of the 20th National Information Systems Security Conference*, pp. 353-365, Baltimore, Maryland, USA, October 1997.

[9] H. Debar, M. Becker and D. Siboni, "A neural network component for an intrusion detection system," *In Proceedings of 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 240-250, Oakland, CA, May 1992.

[10] C. Warrender, S. Forrest and B. Pearlmutter, "Detecting intrusion using calls: Alternative data models," *In Proceedings of IEEE Symposium on Security and Privacy*, pp. 133-145, May 1999.

[11] S.-B. Cho and H.-J. Park, "Efficient anomaly detection by modeling privilege flows with hidden Markov model," *Computers & Security*, vol. 22, no. 1, pp. 45-55, 2003.

[12] R. Lippmann and S. Cunningham, "Improving intrusion detection performance using keyword selection and neural networks," *Computer Networks*, vol. 34, no. 4, pp. 594-603, 2000.

[13] W. Lee, S.J. Stolfo and K.W. Mok, "A data mining framework for building intrusion detection models," *In Proceedings of IEEE Symposium on Security and Privacy*, pp. 120-132, 1999.

[14] A. Lazarevic, L. Ertoz,, V. Kumar, A. Ozgur and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," *In Proceedings of Third SIAM Conference on Data Mining*, May 2003.

[15] W.W. Cohen, "Fast effective rule induction," *In Proceedings of the 12th International Conference on Machine Learning*, pp. 115-123, July 1995.

[16] S.-B. Cho, "Incorporating soft computing techniques into a probabilistic intrusion detection system," *IEEE Trans. on Systems, Man and Cybernetics-Part C:Applications and Reviews*, vol. 32, no. 2, pp. 154-160, May 2002.

[17] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, 1989.

[18] L.R. Rabiner and B.H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, pp. 4-16, January 1986.

[19] H.S. Javitz and A. Valdes, "The SRI IDES statistical anomaly detector," *NIDES Technical Report*, SRI International, 1991.

[20] D.E. Bell and L.J. LaPadula, "Secure computer systems: Unified exposition and multics interpretation," *Mitre Technical Report ESD-TR-75-306*, Mitre Corporation, March 1976.

[21] A.G. Amoroso, *Fundamentals of Computer Security Technology*, PTR Prentice Hall, New Jersy, 1994.

[22] N.A. Macmillan and C.D. Creelman, *Detection Theory: A User's Guide*, Cambridge University Press, Cambridge, 1991.